What is claimed is:

1. A method comprising:

receiving an informational database request; and

determining whether a result set corresponding to said informational database request is stored in a cache, and if so, returning said result set in response to said informational database request, and if not,

sending said informational database request to a database, wherein said database generates said result set, and

determining whether to add said result set to said cache with reference to the cache-worthiness of said result set.

2. The method of claim 1, wherein said cache stores one or more result sets and the database requests corresponding to said one or more result sets, and wherein said determining whether said result set is stored in said cache comprises comparing said database request with said database requests stored in said cache.

3. The method of claim 2, wherein said result set is determined to be stored in said cache if said received database request is identical to one of said database requests stored in said cache.

4. The method of claim 2, wherein said database request comprises a parameter and a query string, and wherein said result set is determined to be stored in said cache if said parameter and said query string are identical to the parameter and query string corresponding to one of said database requests stored in said cache.

5. The method of claim 2, wherein said result set is determined to be stored in said cache if said database request is logically the same as one of said database requests stored in said cache.

6. The method of claim 1, further comprising:

receiving a transactional database request, wherein said transactional database request targets one or more objects within said database; and

invalidating one or more result sets stored in said cache that include data from said one or more objects.

7. The method of claim 1, further comprising:

collecting cache-worthiness data for said result set; and

determining a cache-worthiness value with reference to said cache-worthiness data, wherein said cache-worthiness value is reflective of the cache-worthiness of said result set.

8. The method of claim 7, wherein said cache-worthiness data includes data that is reflective of a number of times said result set has been returned in response to informational database requests.

9. The method of claim 8, wherein said cache-worthiness data further includes data that is reflective of the size of said result set.

10. The method of claim 9, wherein said cache-worthiness data further includes data that is reflective of the amount of time required for said database to generate said result set.

11. The method of claim 10, wherein said cache-worthiness data further includes data that is reflective of the number of times said result set has been invalidated.

12. The method of claim 10, further comprising degrading said cache-worthiness data.

13. The method of claim 12, wherein said degrading comprises degrading said cache-worthiness data on a timed basis.

14. The method of claim 12, wherein said degrading comprises degrading said cache-worthiness data responsive to a miss on said cache.

15. The method of claim 7, wherein said cache-worthiness data includes time data that is reflective of the average time to execute and fetch said result set, and wherein said method further comprises degrading said time data according to

avgTime = avgTime + ((newTime - avgTime) / hit )

wherein avgTime is an average time to execute and fetch said result set, newTime is the most recent measurement of the time to execute and fetch said result set, and hit is the current count of the number of times said result set is requested.

16. The method of claim 7, wherein said cache-worthiness value is determined according to:

cache-worthiness value = (hit / invalid+1) * time

wherein hit is the number of times said result set is requested, invalid is the number of times the result set is invalidated, and time is the average time required to execute and fetch said result set.

17. A result set cache comprising:

first program code means to receive an informational database request; and

second program code means to determine whether a result set corresponding to said informational database request is stored in the result set cache, and if so, to return said result set in response to said informational database request, and if not,

to send said informational database request to a database, wherein said database generates said result set, and

to determine whether to add said result set to the result set cache with reference to the cache-worthiness of said result set.

18. The result set cache of claim 17, further comprising a memory to stores one or more result sets and the database requests corresponding to said one or more result sets, and wherein said second program code means comprises program code means to compare said database request with said database requests stored in said cache.

19. The result set cache of claim 18, wherein said result set is determined to be stored in said cache if said database request is identical to one of said database requests stored in said cache.

20. The result set cache of claim 18, wherein said database request comprises a parameter and a query string, and wherein said result set is determined to be stored in said cache if said parameter and said query string are identical to the parameter and query string corresponding to one of said database requests stored in said cache.

21. The result set cache of claim 18, wherein said result set is determined to be stored in said cache if said database request is logically the same as one of said database requests stored in said cache.

22. The result set cache of claim 17, further comprising:

third program code means to receive a transactional database request, wherein said transactional database request targets one or more objects within said database; and

fourth program code means to invalidate one or more result sets stored in said cache that include data from said one or more objects.

23. The result set cache of claim 17, further comprising:

fifth program code means to collect cache-worthiness data for said result set; and

sixth program code means to determine a cache-worthiness value with reference to said cache-worthiness data, wherein said cache-worthiness value is reflective of the cache-worthiness of said result set.

24. The result set cache of claim 23, wherein said cache-worthiness data includes data that is reflective of a number of times said result set has been returned in response to informational database requests.

25. The result set cache of claim 24, wherein said cache-worthiness data further includes data that is reflective of the size of said result set.

26. The result set cache of claim 25, wherein said cache-worthiness data further includes data that is reflective of the amount of time required for said database to generate said result set.

27. The result set cache of claim 26, wherein said cache-worthiness data further includes data that is reflective of the number of times said result set has been invalidated.

28. The result set cache of claim 17, further comprising seventh program code means to degrade said cache-worthiness data.

29. The result set cache of claim 28, wherein said seventh program code means executes on a timed basis.

30. The result set cache of claim 28, wherein said seventh program code means executes responsive to a miss on the result set cache cache.

31. A result set cache comprising:

memory to store one or more result sets and metadata associated with each of said result sets;

first program code means to collect cache-worthiness data associated with said one or more result sets;

second program code means to determine a cache-worthiness value for each of said one or more result sets, wherein said cache-worthiness values are determined with reference to said cache-worthiness data; and

third program code means to update the contents of the result set cache based at least in part on said cache-worthiness values.

32. The result set cache of claim 31, wherein said third program code means updates the contents of the result set cache on a timed basis.

33. The result set cache of claim 31, wherein said third program code means updates the contents of the result set cache on an as-needed basis.

34. The result set cache of claim 31, wherein said one or more result sets comprise result sets that have been completely fetched.

35. The result set cache of claim 31, wherein said third program code means uses a cache victimization strategy to update the contents of the result set cache.

36. The result set cache of claim 31, further comprising fourth program code means to degrade said cache-worthiness data.

37. The result set cache of claim 36, wherein said third program code means updates the contents of the result cache responsive to said fourth program code means degrading said cache-worthiness data.

38. The result set cache of claim 36, wherein said third program code means and said fourth program code means are executed responsive to a database request being received by the result set cache.

39. The result set cache of claim 36, wherein said third program code means and said fourth program code means are executed responsive to a miss on the result set cache.

40. The result set cache of claim 36, wherein said third program code means and said fourth program code means are executed on a timed basis.

41. The result set cache of claim 31, further comprising fifth program code means to remove result sets stored in said memory responsive to said third program code means.

42. The result set cache of claim 41, wherein said fifth program code means uses a best-fit algorithm to remove result sets stored in said memory.

43. A system comprising:

a database;

an application;

a result set cache including:

first program code means to receive an informational database request from said application; and

second program code means to determine whether a result set corresponding to said informational database request is stored in said result set cache, and if so, to return said result set to said application in response to said informational database request, and if not,

to send said informational database request to said database, wherein said database generates said result set, and

to determine whether to add said result set to said result set cache based at least in part on the cache-worthiness of said result set.

44. The system of claim 43, further comprising a cache driver, wherein said application calls said cache driver to send said informational database request to said result set cache.

45. The system of claim 44, further comprising a database driver, wherein said result set cache calls said database driver to send said informational database request to said database.

46. The system of claim 43, wherein said application and said result set cache use client-side resources, and wherein said database uses server-side resources.

47. The system of claim 43, wherein said application uses client-side resources, and wherein said result set cache and said database use server-side resources.

48. The system of claim 43, wherein said application uses client-side resources, said database uses server-side resources, and said result set cache comprises a stand-alone appliance.

49. A system comprising:

a plurality of clients;

a database;

a result set cache configured to store result sets, wherein said result sets are generated by said database in response to database requests issued by said clients, and wherein said result sets are stored separately for each client.

50. The system of claim 49, wherein a key is associated with each of said result sets, and wherein said key is generated with reference to a client identifier and a query string.

51. The system of claim 49, wherein said client identifier comprises actual user information.

52. The system of claim 49, wherein said result set cache restricts access to said result sets such that said clients are only able to access their own separately stored result sets.

53. The system of claim 49, wherein said result set cache comprises first program code means for translating database requests received from said clients into canonical form, wherein a key is associated with each of said result sets, said key being generated in canonical form with reference to a client identifier and a query string.

54. A method of maintaining consistency in a result set cache, wherein said cache stores one or more result sets, said method comprising:

receiving a database request;

determining whether said database request is informational or transactional; and

if said database request is transactional, invalidating result sets stored in the result set cache that include data targeted by said database request.

55. The method of claim 54, wherein each of said result sets is generated based on one or more objects stored in a database, and wherein said invalidating comprises:

parsing said database request to determine whether any of said one or more objects are affected by said database request; and

marking those results sets that were generated based on the affected objects.

56. The method of claim 54, further comprising invalidating result sets stored in the result set cache on a timed basis.